IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants: SCHLEISS et al.

Serial No.: 09/902,201

Title: Transactional Data Communications for Process Control Systems

Filed: July 10, 2001

Group Art Unit: 2143

Examiner: Nguyen, Phuoc H.

Atty. Docket Number: 06005/37169

Customer No.: 04743

I hereby certify that this paper is being deposited with the United States Postal Service, first class postage prepaid, addressed to: Mail Stop RCE, Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450, on the date indicated:

MAY 9, 2006
Date

Aaron M. Peters
Registration No. 48,801
Attorney for Applicants

DECLARATION OF MARK NIXON PURSUANT TO 37 C.F.R. § 1.131

Mail Stop RCE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Dear Sir:

Mark Nixon hereby states as follows:

1. I am a named co-inventor of the subject matter claimed in the above-identified patent application ("the patent application").

2. I make this declaration for the purpose of providing evidence that the system or method for communicating transactional process control information, as recited in the claims of the patent application, was in our possession at least as early as the May 24, 2001 filing date of Dodge et al., U.S. Patent No. 6,795,778.

3. Attached hereto as Attachment A is a document dated at least as early as May 24, 2001. Portions of the attachment have been redacted, including redactions to remove date information. Page numbers have been added for convenience.

4. Attachment A is entitled "DeltaV EasyIT" and illustrates an example of a method of communicating information within an enterprise having a process control system and a plurality of information technology systems, as recited by claim 1.

5. Attachment A was prepared in the United States and has been maintained as a business record in the normal course of business.

6. Attached hereto as Attachment B is a document dated at least as early as May 24, 2001. Portions of the attachment have been redacted, including redactions to remove date information.

7. Attachment B is entitled "Enterprise Optimization Is Here At Last – Answering Today's Problems with OPC, XML, Biztalk, and Easy-IT" and illustrates an example of a method of communicating information within an enterprise having a process control system and a plurality of information technology systems, as recited by claim 1.

8. Attachment B was prepared in the United States and has been maintained as a business record in the normal course of business.

9. Pages 33 and 69 of Attachment A and pages 9 and 18 of Attachment B illustrate that transactional process control information (e.g., device alert information) was generated. The device alert information related to a transactional event within a process control system (e.g., maintenance alert for a plugged sensor of an intelligent field device).

10. Pages 35 and 51-54 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the transactional process control information (e.g., device alert information) was formatted based on a first extensible markup language schema (e.g., device alert schema, input schema; XML schema; inbound schema) to form formatted transactional process control information.

11. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that the formatted transactional process control information was sent to a transactional information server (e.g., BizTalk Server; XML transaction server) via a web services interface (e.g., different transport services are supported including HTTP, etc.; data communication over the World Wide Web; Internet communications).

12. Pages 54-57 and 73-75 of Attachment A and pages 16-18 of Attachment B illustrate that the formatted transactional process control information was mapped to a second extensible markup language schema (e.g., output schema; transformation of an inbound schema to an outbound schema) associated with one of a plurality of information technology systems (e.g., maintenance system; Computerized Maintenance Management System (CMMS)) to form mapped transactional process control information.

13. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9, 10 and 16-18 of Attachment B illustrate that the mapped transactional process control information was sent to a first one of the plurality of information technology systems (e.g., maintenance system; CMMS) to use the mapped transactional process control information to perform a function related to the transactional event (e.g., work order).

14. Attachments A and B demonstrate our possession of a method of communicating information within an enterprise having a process control system and a plurality of information technology systems, as recited by claim 1 in the above identified patent application, at least as early as May 24, 2001.

15. Attachments A and B also depict an example of a system for use in an enterprise having a plurality of information technology systems, as recited by claim 10.

16. Pages 35, 47, 51-57 and 70 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that a process control system (e.g., control system; automation system) was adapted to format transactional process control information (e.g., device alert information) based on an extensible markup language (e.g., XML) and a plurality of input schemas (e.g., device alert schema; input schema; XML schema; inbound schema). Each of the plurality of input schemas was associated with a type of transactional process control information related to a transactional event within the process control system (e.g., device alert schema).

17. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that a web services interface was communicatively coupled to the process control system (e.g., different transport services are supported including HTTP, etc.; data communication over the World Wide Web; Internet communications).

18. Pages 46, 47, 49, 50, 54-57, 67 and 73-75 of Attachment A and pages 9-11, 13, 14 and 16-18 of Attachment B illustrate a transactional data server (e.g., BizTalk Server; XML transaction server). The transactional data server was communicatively coupled to the web services interface (e.g., different transport services are supported including HTTP, etc.; data communication over the World Wide Web; Internet communications) and a plurality of information technology systems (e.g., maintenance system; CMMS). The transactional data server was adapted to map transactional process control information that had been formatted based on the extensible markup language and the plurality of input schemas to a plurality of output schemas (e.g., output schema, transformation of an inbound schema to an outbound

schema). Each of the plurality of output schemas was associated with an application that is executed within one of the plurality of information technology systems (e.g., BizTalk server maps data between applications; CMMS application). The transactional data server was further adapted to send mapped transactional process control information to one of the plurality of information technology systems (e.g., maintenance system, CMMS) to use the mapped transactional process control information to perform a function related to the transactional event (e.g., work order).

19. Attachments A and B demonstrate our possession of a system for use in an enterprise having a plurality of information technology systems, as recited by claim 10 in the above identified patent application, at least as early as May 24, 2001.

20. Attachments A and B also depict an example of a method of processing transactional process control data, as recited by claim 17.

21. Pages 35 and 51-54 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that transactional process control data (e.g., device alert information) was wrapped in an XML wrapper (e.g., device alert schema, input schema, XML schema, inbound schema) to form XML wrapped transactional process control data related to a transactional event within the process control system (e.g., maintenance alert for a plugged sensor).

22. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that the XML wrapped transactional process control data was sent via a web services interface and a communication network (different transport services are supported including HTTP, etc.; Internet communications) to an XML data server (e.g., BizTalk Server; XML transaction server).

23. Pages 54-57 and 73-75 of Attachment A and pages 16-18 of Attachment B illustrate that the XML wrapped transactional process control data was mapped to an XML output schema (e.g., output schema; transformation of an inbound schema to an outbound schema) associated with one of a plurality of information systems (e.g., maintenance system, CMMS) that were communicatively coupled to the communication network (e.g., Internet) to form mapped XML transactional process control data.

24. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9, 10, 16-18 of Attachment B illustrate that the mapped XML transactional process control data was sent to the one of the plurality of information systems (e.g., maintenance system, CMMS) via the

communication network (e.g., Internet) to use the mapped transactional process control data to perform a function related to the transactional event (e.g., work order).

25. Attachments A and B demonstrate our possession of a method of processing transactional process control data, as recited by claim 17 in the above identified patent application, at least as early as May 24, 2001.

26. Attachments A and B further depict an example of a method of processing transactional process control data, as recited by claim 22.

27. Pages 35 and 51-54 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the transactional process control data (e.g., device alert information) was encapsulated in a markup language wrapper (e.g., device alert schema; input schema; XML schema; inbound schema) to form encapsulated transactional process control data related to a transactional event (e.g., device alert) within a process control system.

28. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that the encapsulated transactional process control data was sent via a web services interface and a communication network (e.g., different transport services are supported including HTTP, etc.; Internet communications) to a markup language data server (e.g., BizTalk Server; XML transaction server).

29. Pages 6, 54-57 and 73-75 of Attachment A and pages 5, 16-18 of Attachment B illustrate that the encapsulated transactional process control data was mapped to an output schema (e.g., output schema, transformation of an inbound schema to an outbound schema) associated with one of an enterprise resource planning system (e.g., ERP) and a manufacturing execution system (e.g., MES, CMMS) to form mapped transactional process control data.

30. Pages 6, 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 5, 9, 10 and 16-18 of Attachment B illustrate that the mapped transactional process control data was sent to the one of the enterprise resource planning system (e.g., ERP) and the manufacturing execution system (e.g., MES; CMMS) to use the mapped transactional process control data to perform a function related to the transactional event (e.g., work order).

31. Attachments A and B demonstrate our possession of a method of processing transactional process control data, as recited by claim 22 in the above identified patent application, at least as early as May 24, 2001.

32. Attachments A and B further depict an example of a method of communicating transactional process control information within an enterprise, as recited by claim 27.

33. Pages 35 and 51-54 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the transactional process control information (e.g., device alert information) was formatted based on a first extensible markup language schema (e.g., device alert schema, input schema, XML schema; inbound schema) to form formatted transactional process control information related to a transactional event (e.g., maintenance alert for a plugged sensor of an intelligent field device).

34. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that the formatted transactional process control information was sent to a transactional information server (e.g., BizTalk Server; XML transaction server).

35. Pages 54-57 and 73-75 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the formatted transactional process control information was mapped to a second extensible markup schema associated with a process control system (e.g., automated system; plant) to form mapped transactional process control information (e.g., transactions at the enterprise level could use the application to send XML based information to the plant; bi-directional).

36. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9-11, 13, 14 and 16-18 of Attachment B illustrate that the mapped transactional process control information was sent to the process control system (e.g., automated system; plant) via a web services interface (e.g., different transport services are supported including HTTP, etc.; Internet communications) to use the mapped transactional process control information to perform a function related to the transactional event (e.g., work order).

37. Attachment A demonstrates our possession of a method of communicating transactional process control information within an enterprise, as recited by claim 27 in the above identified patent application, at least as early as May 24, 2001.

38. Attachments A and B further depict an example of a method of processing a device alarm for use within an enterprise including a process control system and a maintenance management system, as recited by claim 31.

39. Pages 35 and 51-54 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the device alarm (e.g., device alert) was formatted based on an XML input schema (e.g., device alert schema, input schema, inbound schema) to form an XML device alarm.

40. Pages 46, 47, 49 and 50 of Attachment A and pages 9-11, 13, 14 and 18 of Attachment B illustrate that the XML device alarm was sent to an XML transaction server (e.g., BizTalk Server; XML transaction server).

41. Pages 54-57 and 73-75 of Attachment A and pages 11 and 16-18 of Attachment B illustrate that the XML device alarm was mapped to an XML output schema (e.g., output schema; transformation of an inbound schema to an outbound schema) associated with the maintenance management system (e.g., maintenance system, CMMS) to form a mapped XML device alarm.

42. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9, 10 and 16-18 of Attachment B illustrate that the mapped XML device alarm was sent to the maintenance management system (e.g., maintenance system, CMMS) to use the mapped XML device alarm to perform a function related to the device alarm (e.g., work order).

43. Attachments A and B demonstrate our possession of a method of processing a device alarm for use within an enterprise including a process control system and a maintenance management system, as recited by claim 31 in the above identified patent application, at least as early as May 24, 2001.

44. Attachments A and B further depict an example of a method of processing equipment condition information for use within an enterprise including a process control system and an information technology system, as recited by claim 32.

45. Pages 35, 51-54 and 64 of Attachment A and pages 10, 11 and 16-18 of Attachment B illustrate that the equipment condition information (e.g., historical data; status of plant equipment; intelligent field device) was formatted based on an XML input schema (e.g., XML schema; inbound schema) to form an XML message.

46. Pages 54-57 and 73-75 of Attachment A and pages 16-18 of Attachment B illustrate that the XML message was mapped to an XML output schema (e.g., output schema, transformation of an inbound schema to an outbound schema) associated with the information technology system (e.g., maintenance system; CMMS) to form a mapped XML message.

47. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9, 10 and 16-18 of Attachment B illustrate that the mapped XML message was sent to the information technology system (e.g., maintenance system; CMMS) to use the mapped XML message to perform a function related to the message (e.g., work order).

48. Attachments A and B demonstrate our possession of a method of processing equipment condition information for use within an enterprise including a process control system and an information technology system, as recited by claim 32 in the above identified patent application, at least as early as May 24, 2001.

49. Attachments A and B further depict an example of a method of processing process condition information for use within an enterprise including a process control system and an information technology system, as recited by claim 33.

50. Pages 35, 47 and 51-54 of Attachment A and pages 10, 11 and 16-18 of Attachment B illustrate that the process condition information (e.g., production control; production schedule data; production data) was formatted based on an XML input schema (e.g., input schema, XML schema, inbound schema) to form an XML message.

51. Pages 54-57 and 73-75 of Attachment A and pages 16-18 of Attachment B illustrate that the XML message was mapped to an XML output schema (e.g., output schema; transformation of an inbound schema to an outbound schema) associated with the information technology system (e.g., maintenance system; CMMS; production scheduling) to form a mapped XML message.

52. Pages 46, 47, 49, 50, 54-57 and 67 of Attachment A and pages 9, 10 and 16-18 of Attachment B illustrate that the mapped XML message was sent to the information technology system (e.g., maintenance system; CMMS; production scheduling) to use the mapped XML message to perform a function related to the message (e.g., work order; production scheduling).

53. Attachments A and Bdemonstrate our possession of a method of processing process condition information for use within an enterprise including a process control system and an information technology system, as recited by claim 33 in the above identified patent application, at least as early as May 24, 2001.

54. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that

these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. §1001 and that such willful false statements may jeopardize the validity of the above-referenced patent application and any patent issued therefrom.

Date: 5/9/06 By: _____

Mark Nixon

THIS PAGE BLANK (USPTO)

DeltaV EasyIT

REDACTED

REDACTED

REDACTED

REDACTED

What is EasyIT?

- → IT = Information Technology
- → The sharing of information with other systems outside of your process automation system
  - ❑ Manufacturing Execution Systems (MES)
  - ❑ Enterprise Resource Planning (ERP)
- → EasyIT is Enterprise Optimization
- → DeltaV makes it Easy! No user programming required!

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

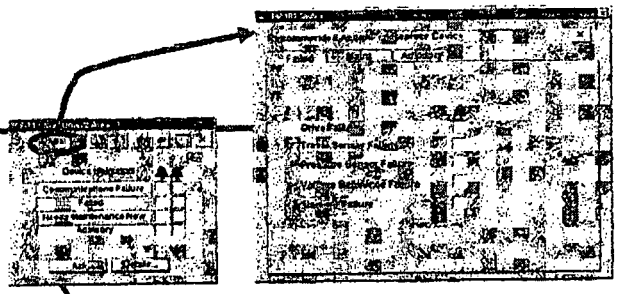
REDACTED

REDACTED

REDACTED

REDACTED

PlantWeb Rules



Device Alert information 3051

Date	REDACTED
Time	02:11:15 PM
Tag	FT-100
Alert	MAINTENANCE
Description	Sensor Plugged
Priority	4
Action	Remove and repair
State	act/unack

This is the data needed from the DeltaV system to pass the device alert to another application

REDACTED

Device Alert Schema - Example

→ Write an XML wrapper around this device alert data and put it in a file

<deviceAlert3051>

<date> REDACTED </date>

<time> 02:11:15 PM </time>

<tag> FT-100 </tag>

<alert> MAINTENANCE </alert>

<description> Sensor Plugged </description>

<priority> 4 </priority>

<action> Remove and repair </action>

<state> act/unack </state>

</deviceAlert3051>

XML

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

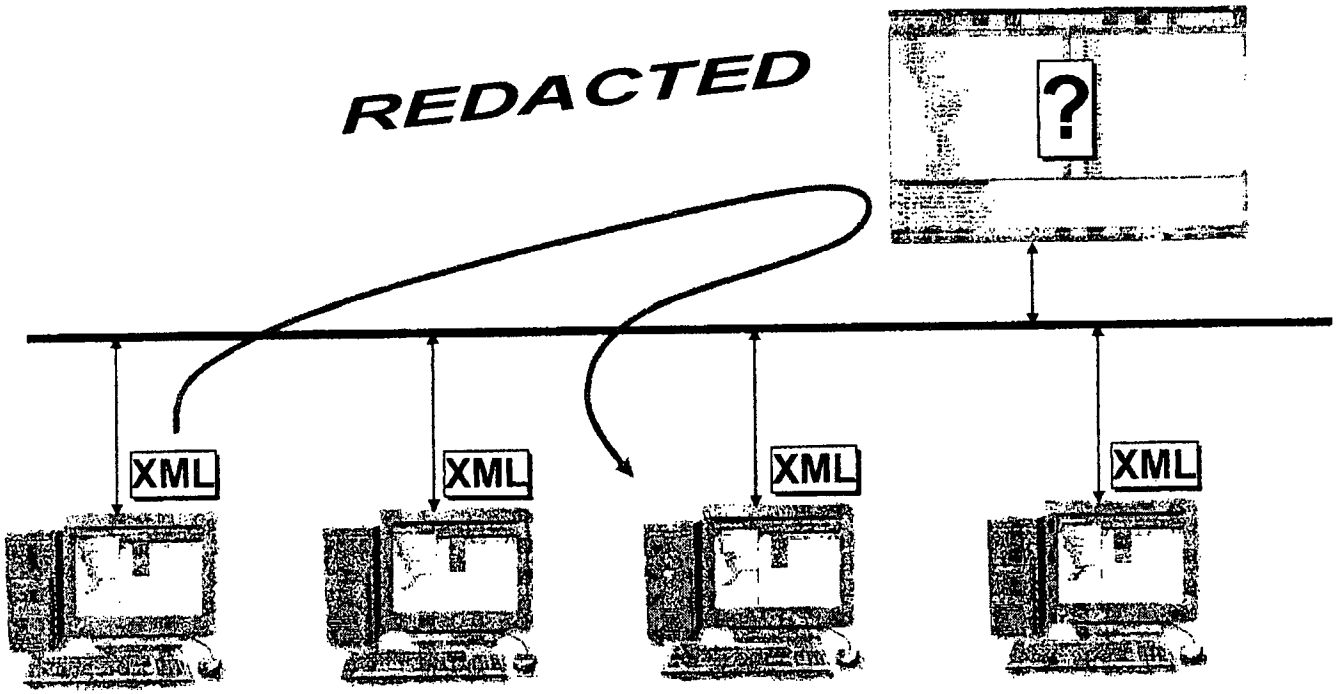
REDACTED

REDACTED

REDACTED

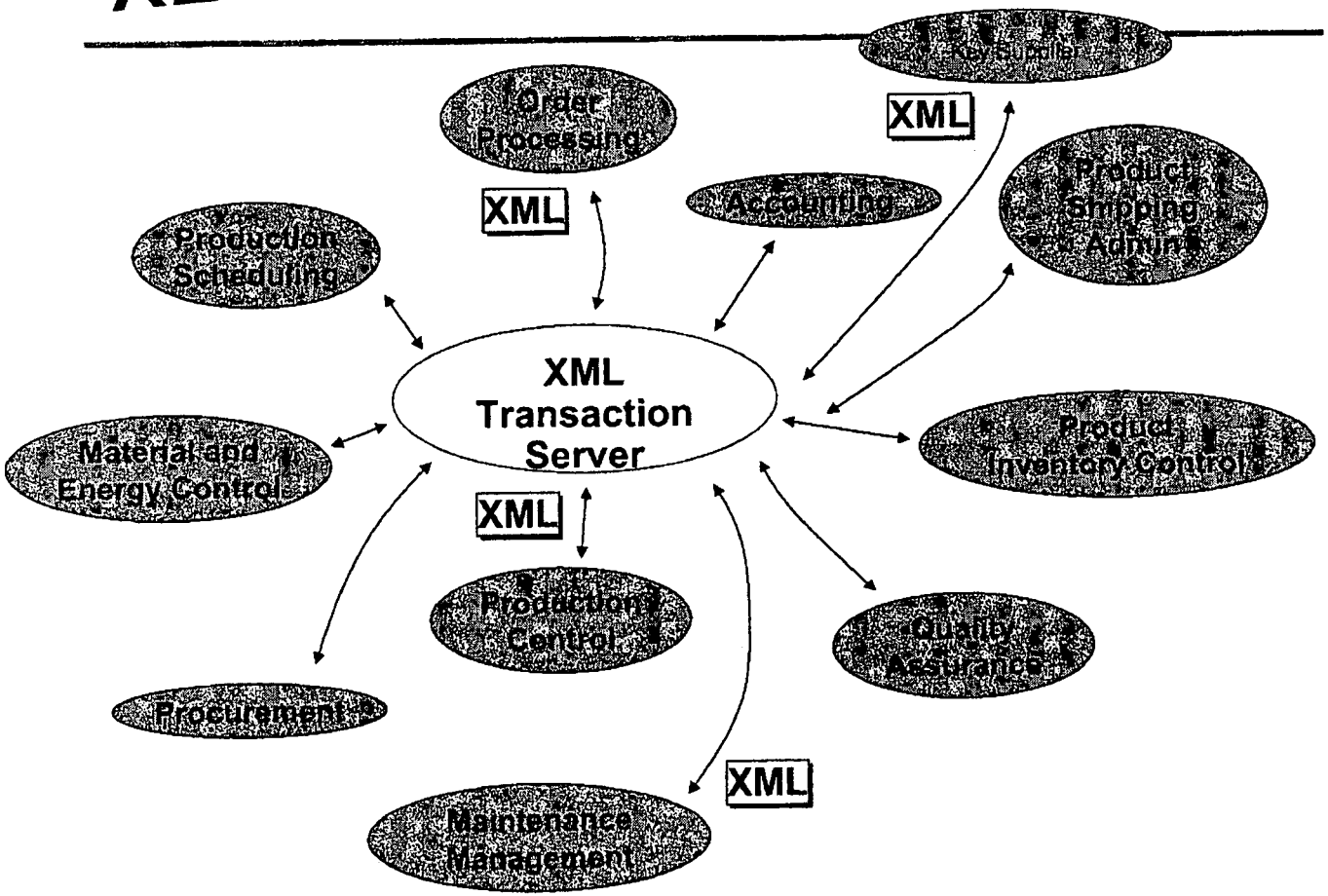
REDACTED

REDACTED



Fisher-Rosemount Systems, Inc.
Slide 46

REDACTED

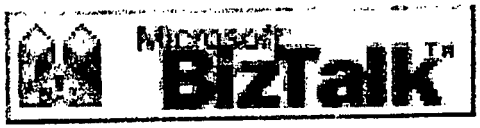


Fisher-Rosemount Systems, Inc.
Slide 47

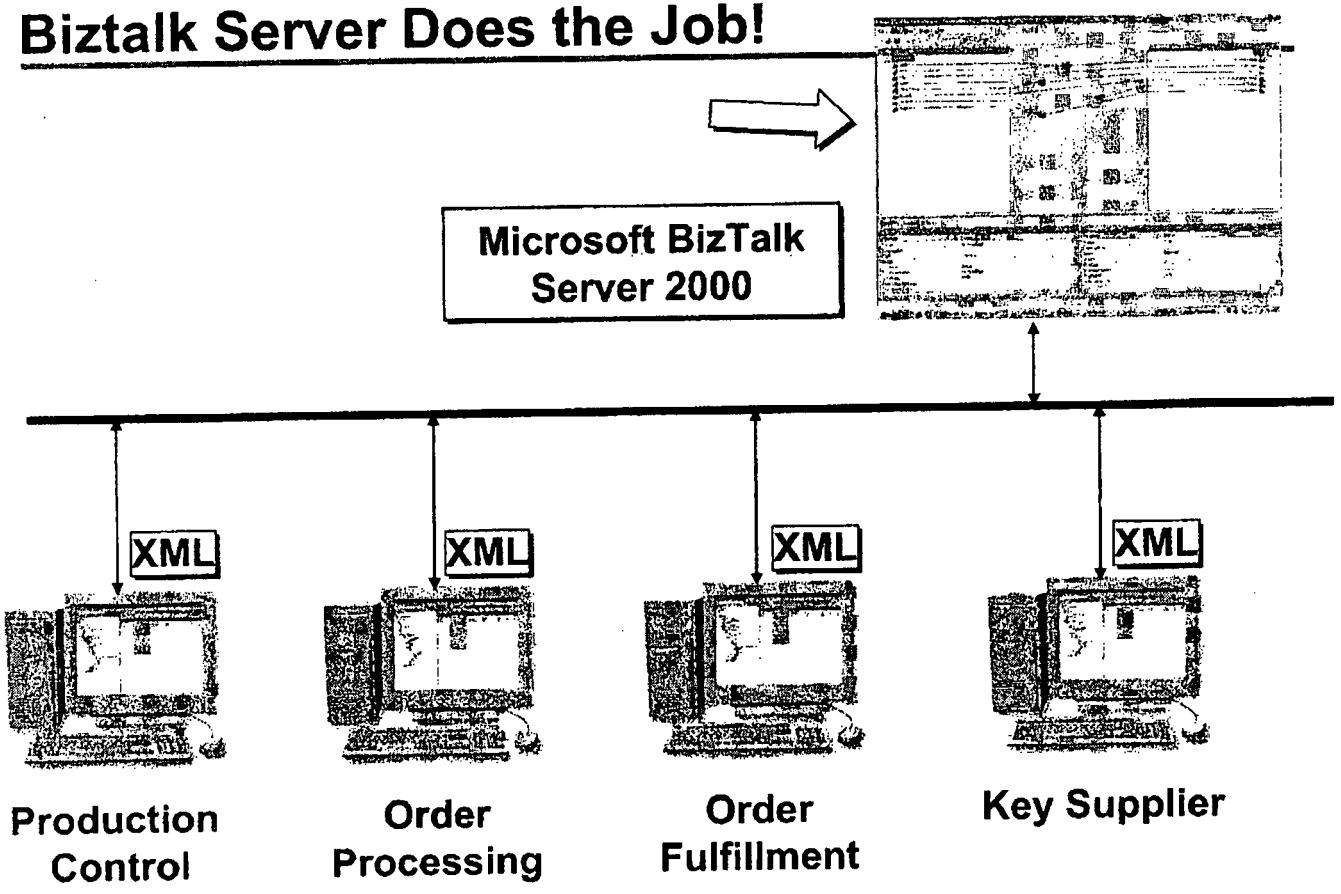
REDACTED

What is BizTalk Server 2000?

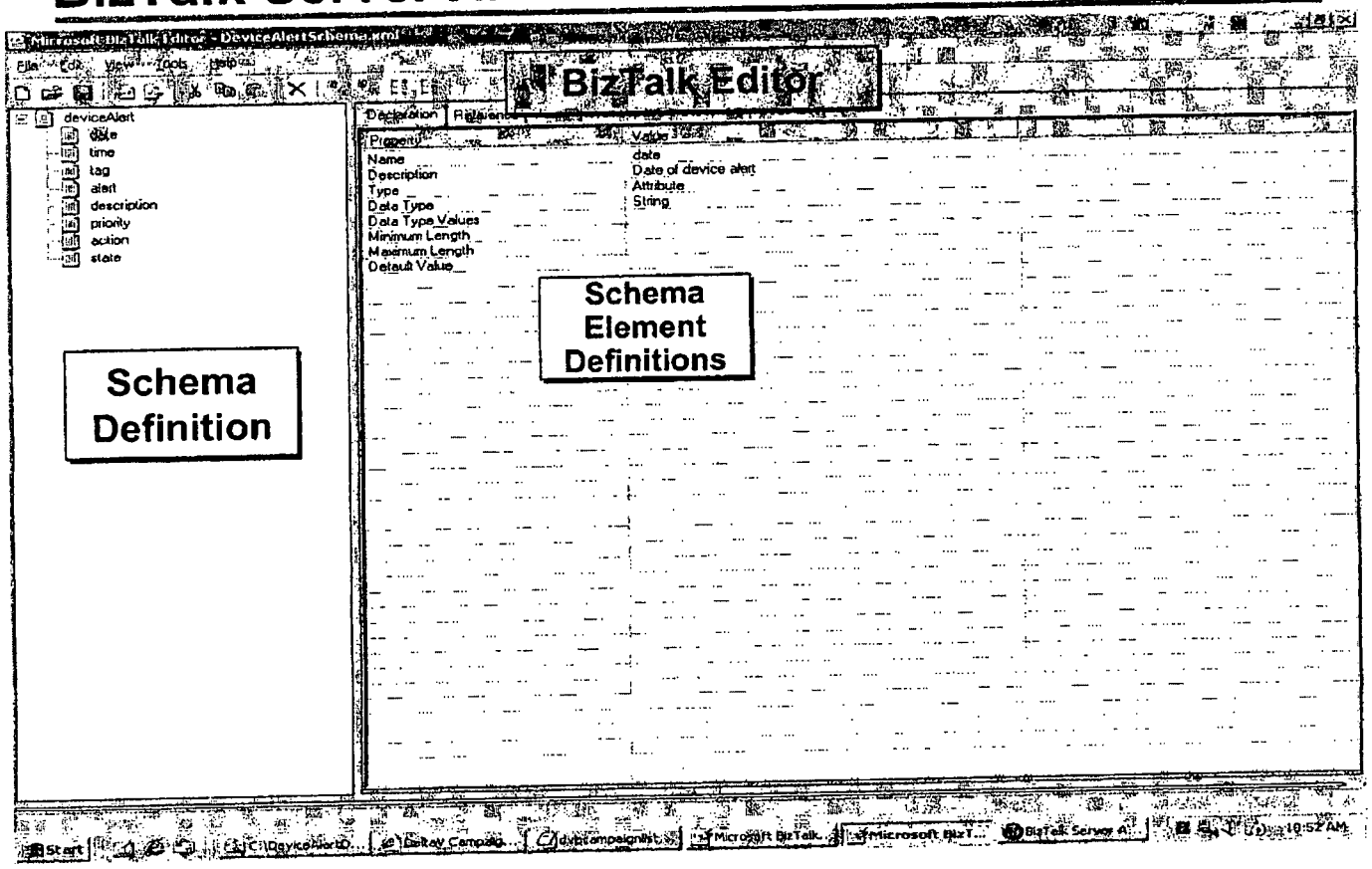
- → Biztalk server facilitates the handling of transactions
- → Many different data types and structures are accommodated including SQL, binary files, XML, etc.
- → Many different transport services are supported including HTTP, HTTPS, SMTP, MSMQ, FTP, flat files, fax
- → BizTalk Server includes tools to vastly simplify the application configuration
- → Guaranteed transactions are ensured
- → Facilitates data communication over the World Wide Web
- → Available REDACTED
- → Makes transaction application setup easy!



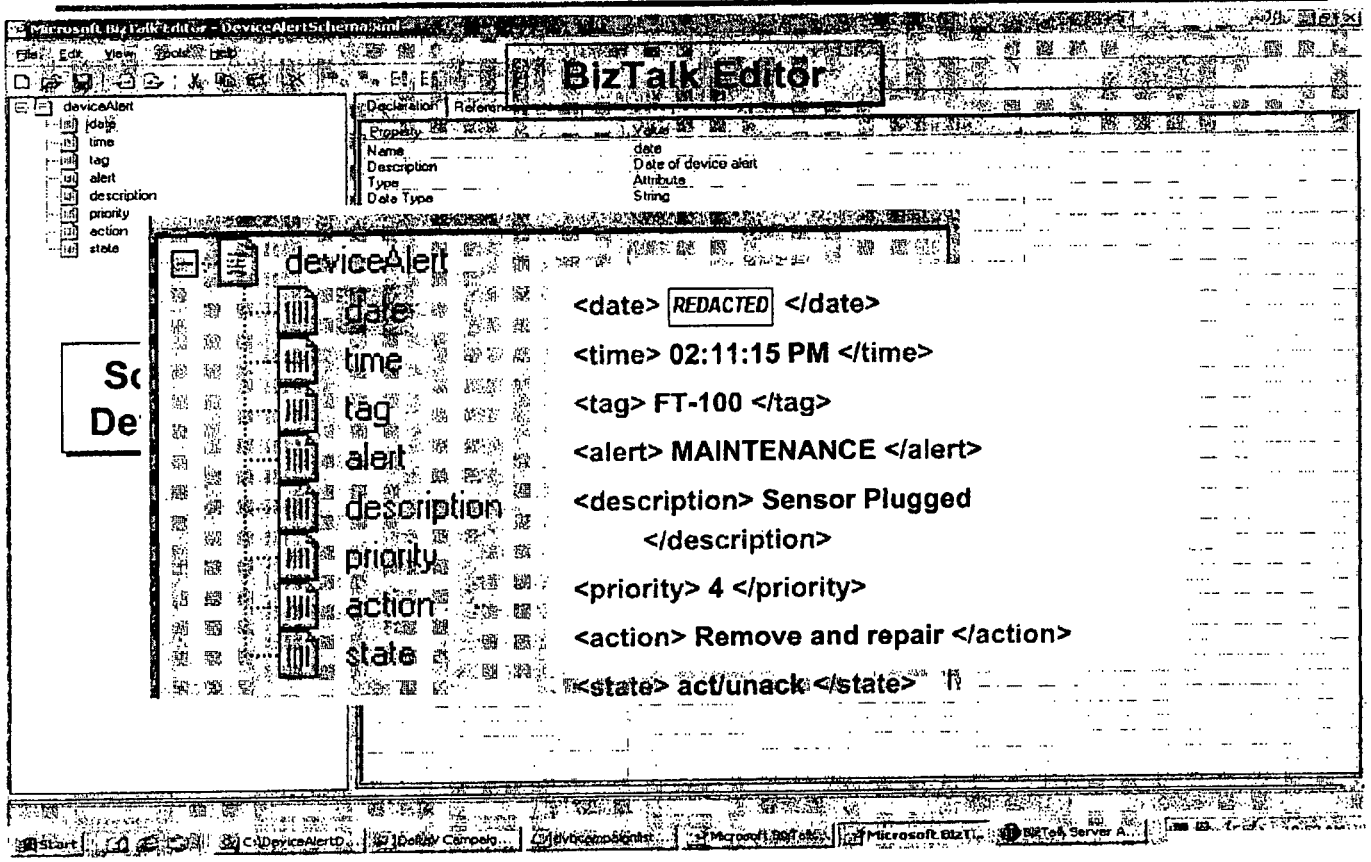
Biztalk Server Does the Job!



BizTalk Server XML Schema Editor



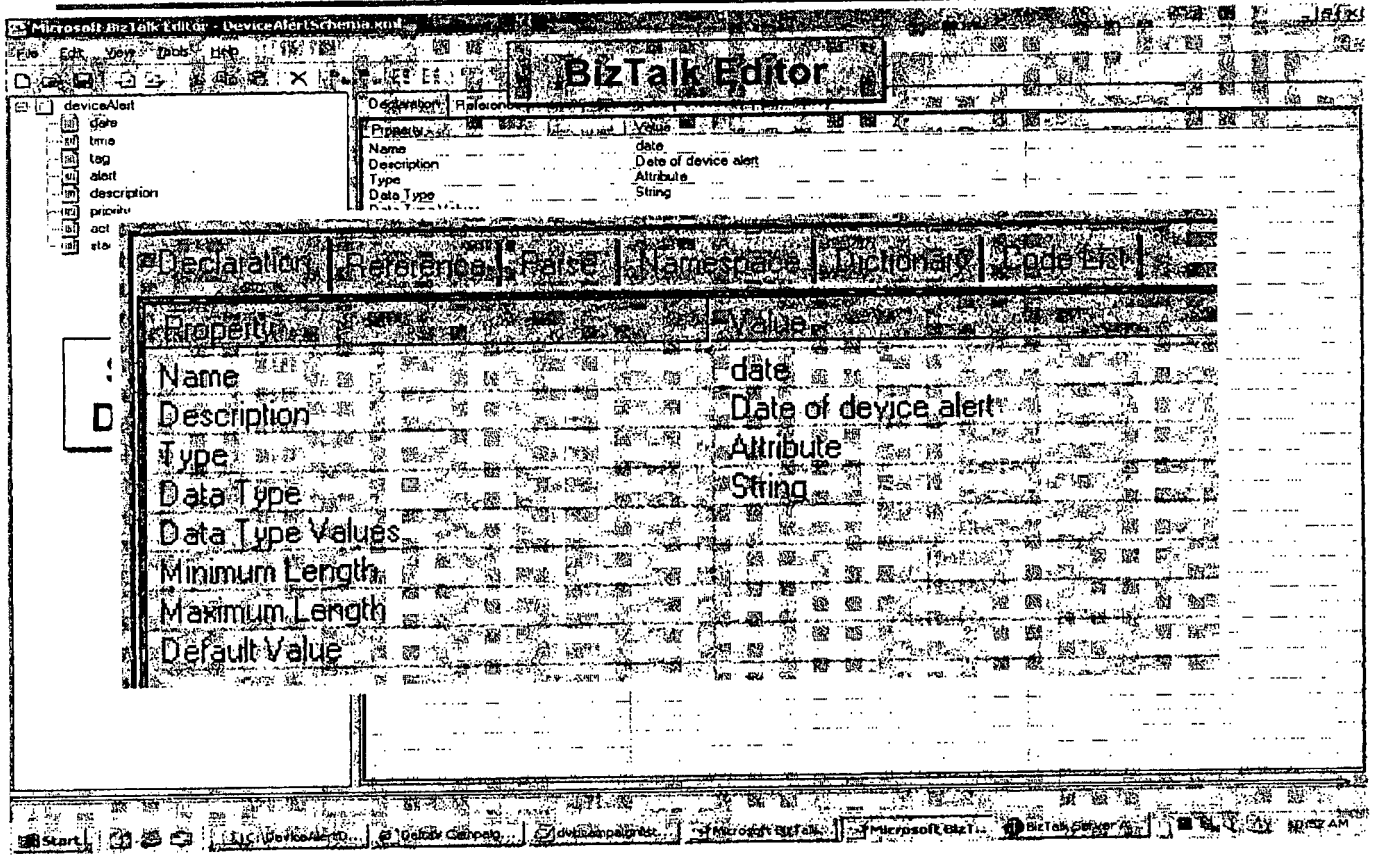
BizTalk Server XML Schema Editor



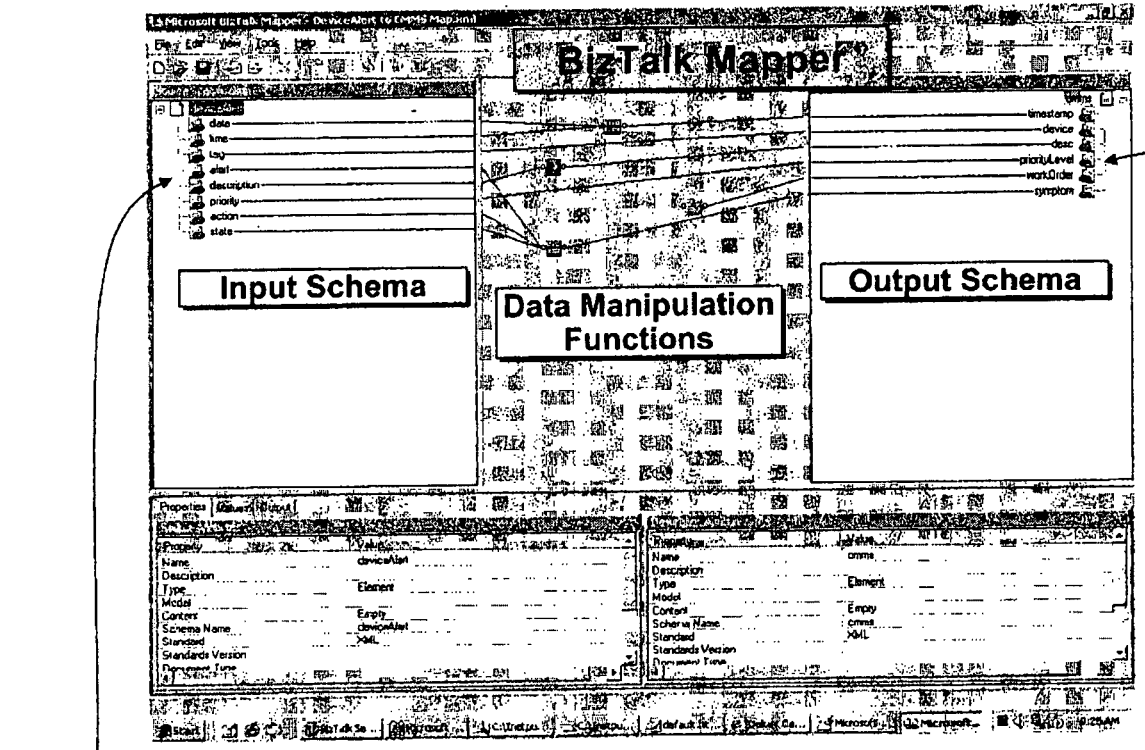
Fisher-Rosemount Systems, Inc.

Slide 52

BizTalk Server XML Schema Editor



BizTalk Server Maps Data Between Applications



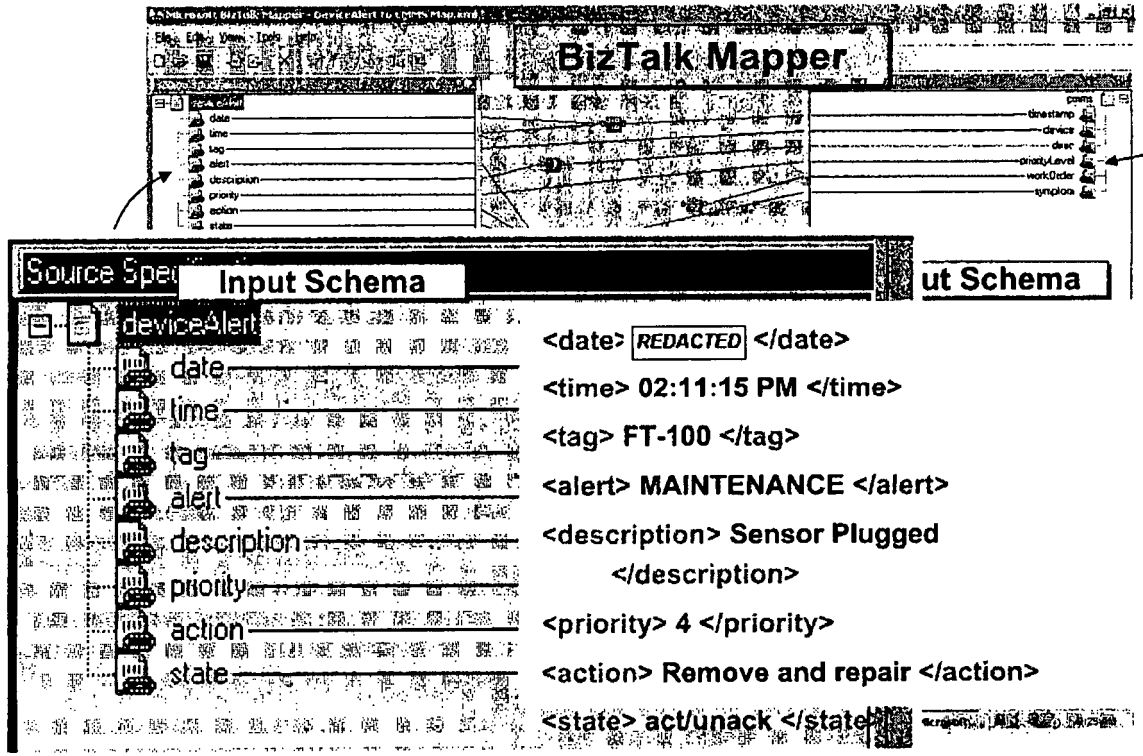
Automation System

Microsoft BizTalk

Maintenance System

Fisher-Rosemount Systems, Inc.
Slide 54

BizTalk Server Maps Data Between Applications

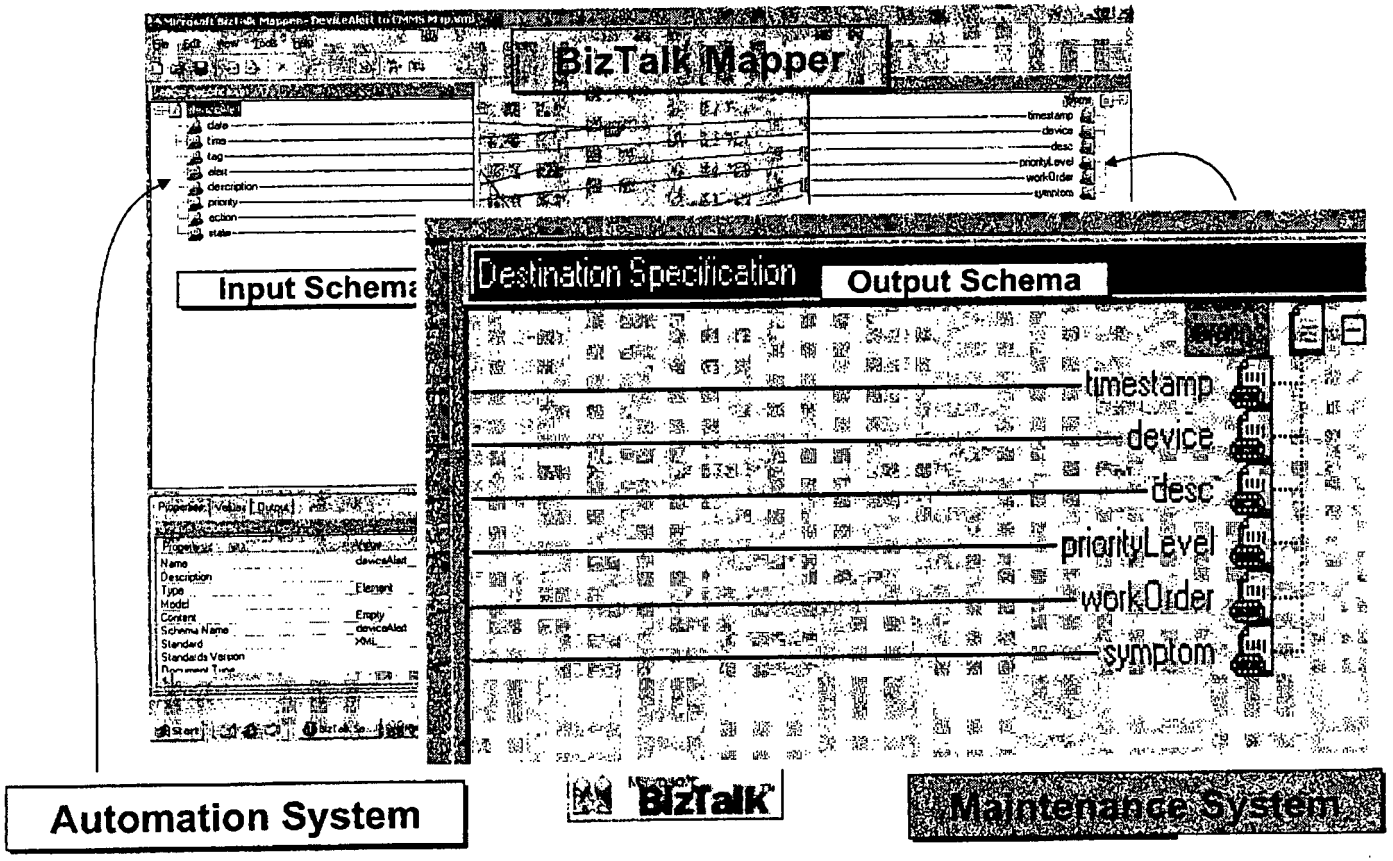


Automation System

BizTalk

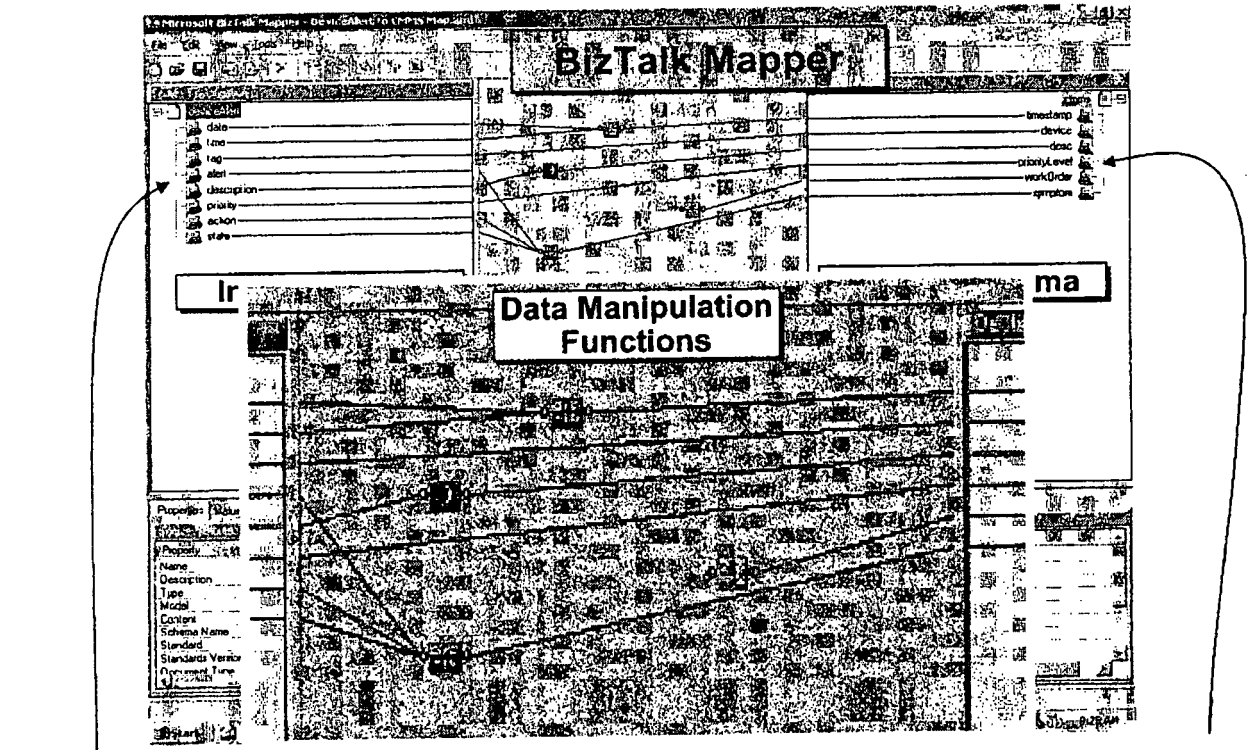
Maintenance System

BizTalk Server Maps Data Between Applications



Fisher-Rosemount Systems, Inc.
Slide 56

BizTalk Server Maps Data Between Applications



Automation System

Microsoft BizTalk

Maintenance System

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

- → The interface with DeltaV for transactional data
- → Based on XML
- → Initial transactions will include
  - ❑ Export batch historical data
  - ❑ Export continuous historical data
  - ❑ Export batch status
  - ❑ Export advanced control data
  - ❑ Export configuration data
  - ❑ Import batch historical data
  - ❑ Import continuous historical data
  - ❑ Import recipe information
  - ❑ Import configuration data
  - ❑ External campaign control

REDACTED

REDACTED

Device Alert Demo

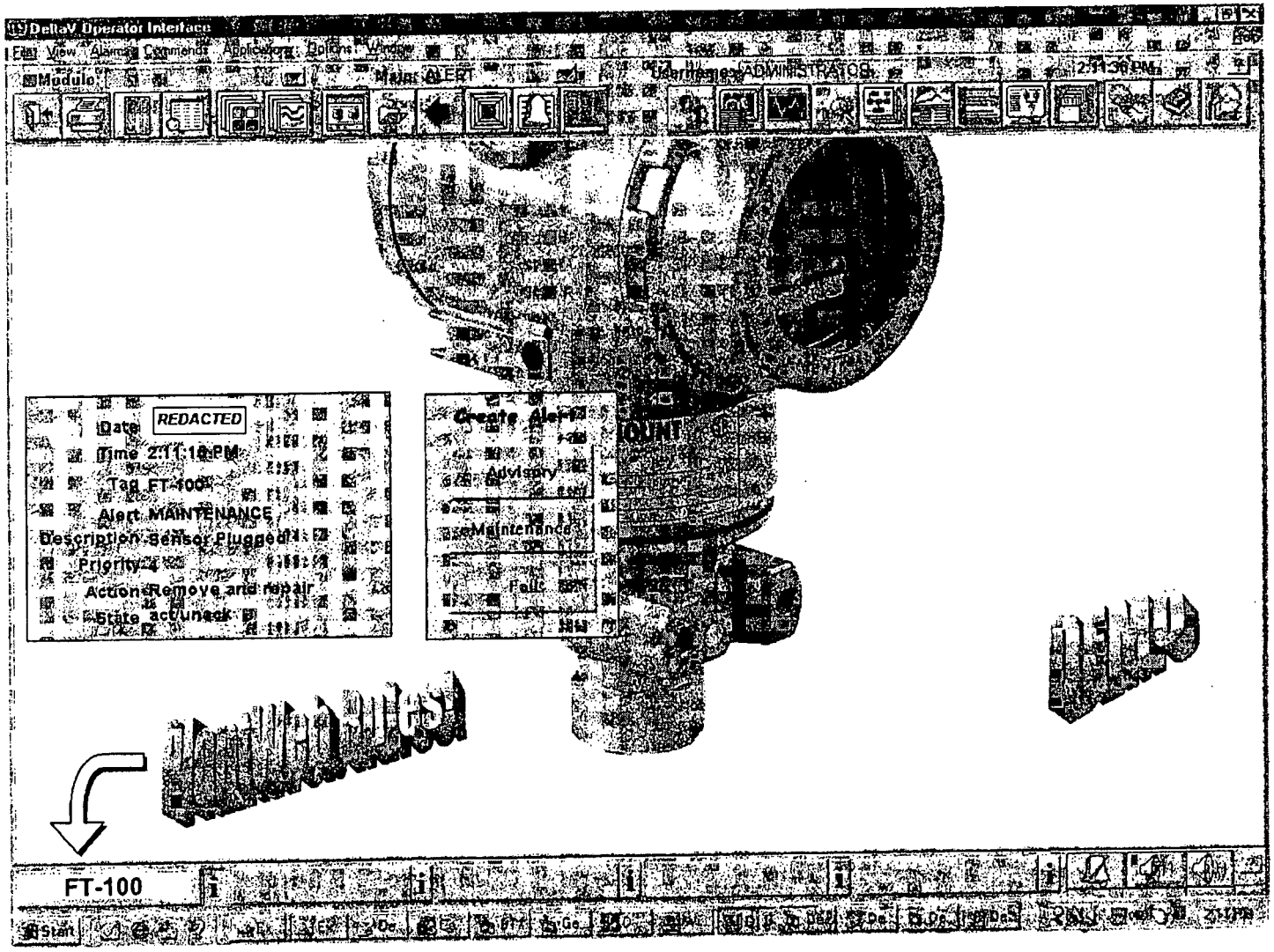
→ On device problem (simulated)-

- ❑ Generate a work order through CMMS
- ❑ Page the maintenance tech
- ❑ Alert the plant operators on their PC screens

→ With Biztalk Server, DeltaV Easy IT, and XML enabled data . . .

REDACTED

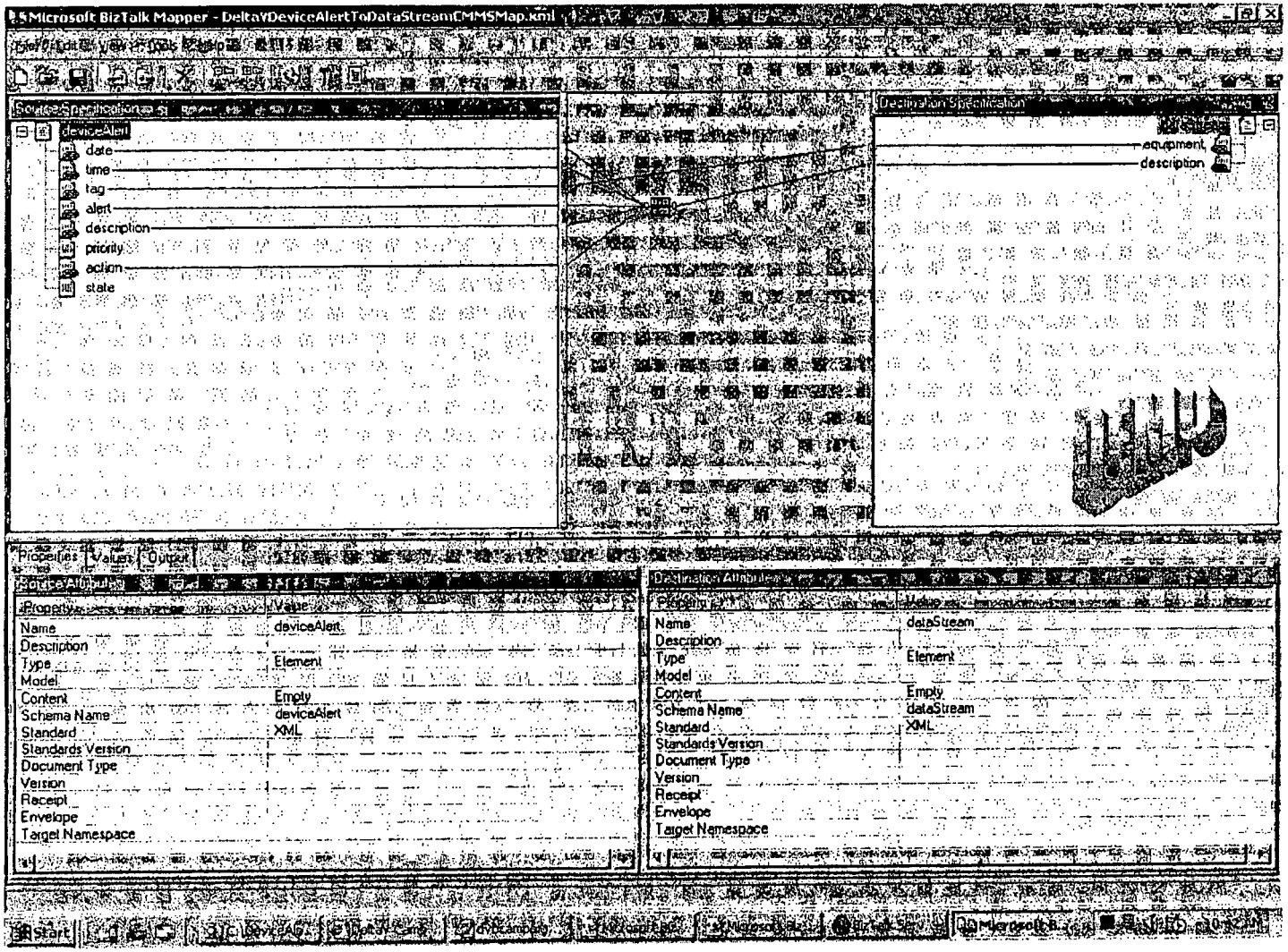
REDACTED

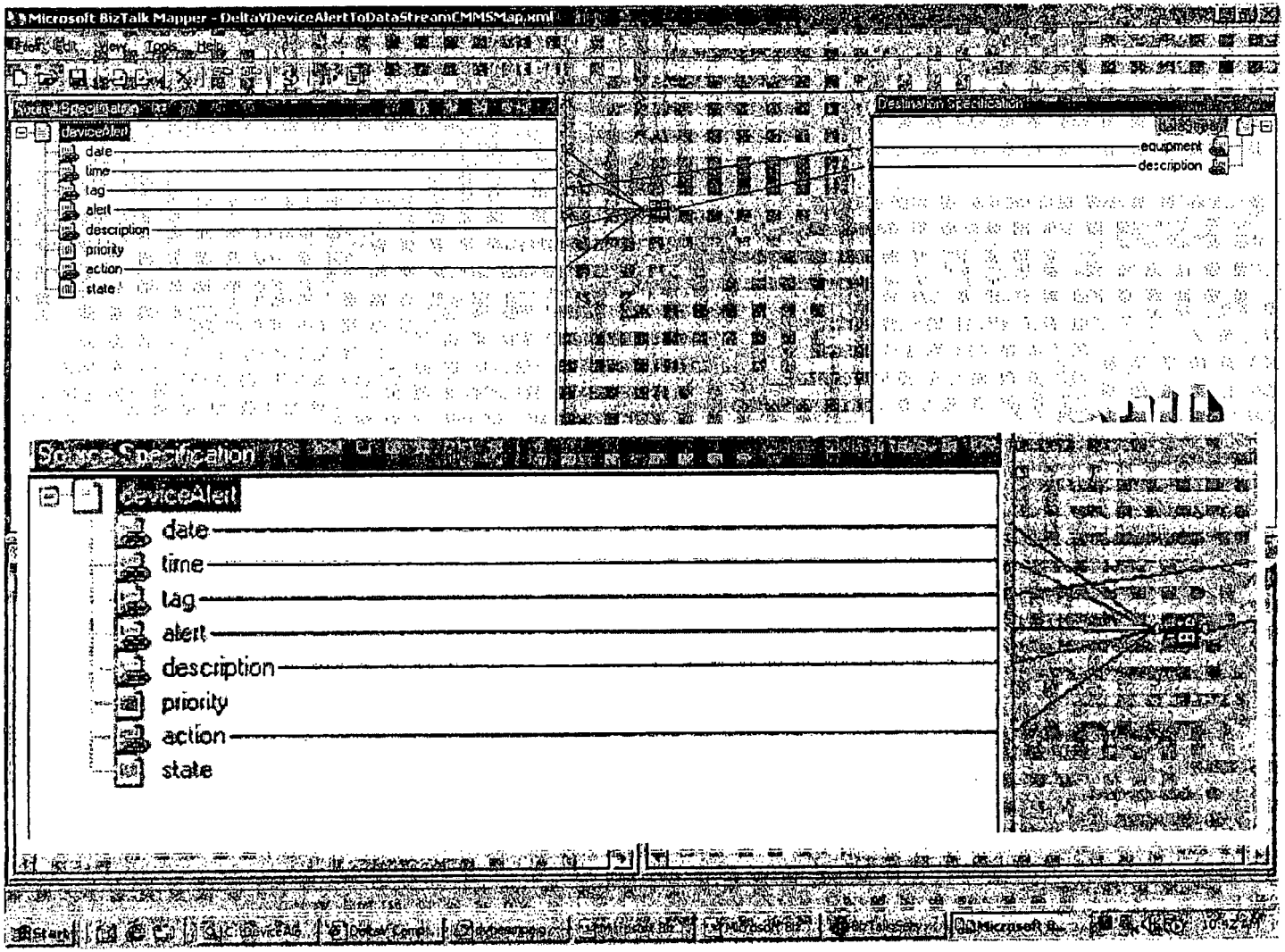

DeltaV Operator Interface
Module
Main: ALERT
User Name: ADMINISTRATOR
REDACTED
Date
Time 2:11:18 PM
Tag FT-100
Alert MAINTENANCE
Description Sensor Plugged
Priority 4
Action Remove and repair
State act/unack
Create Alert
Advisory
Maintenance
Failure
FT-100
Start

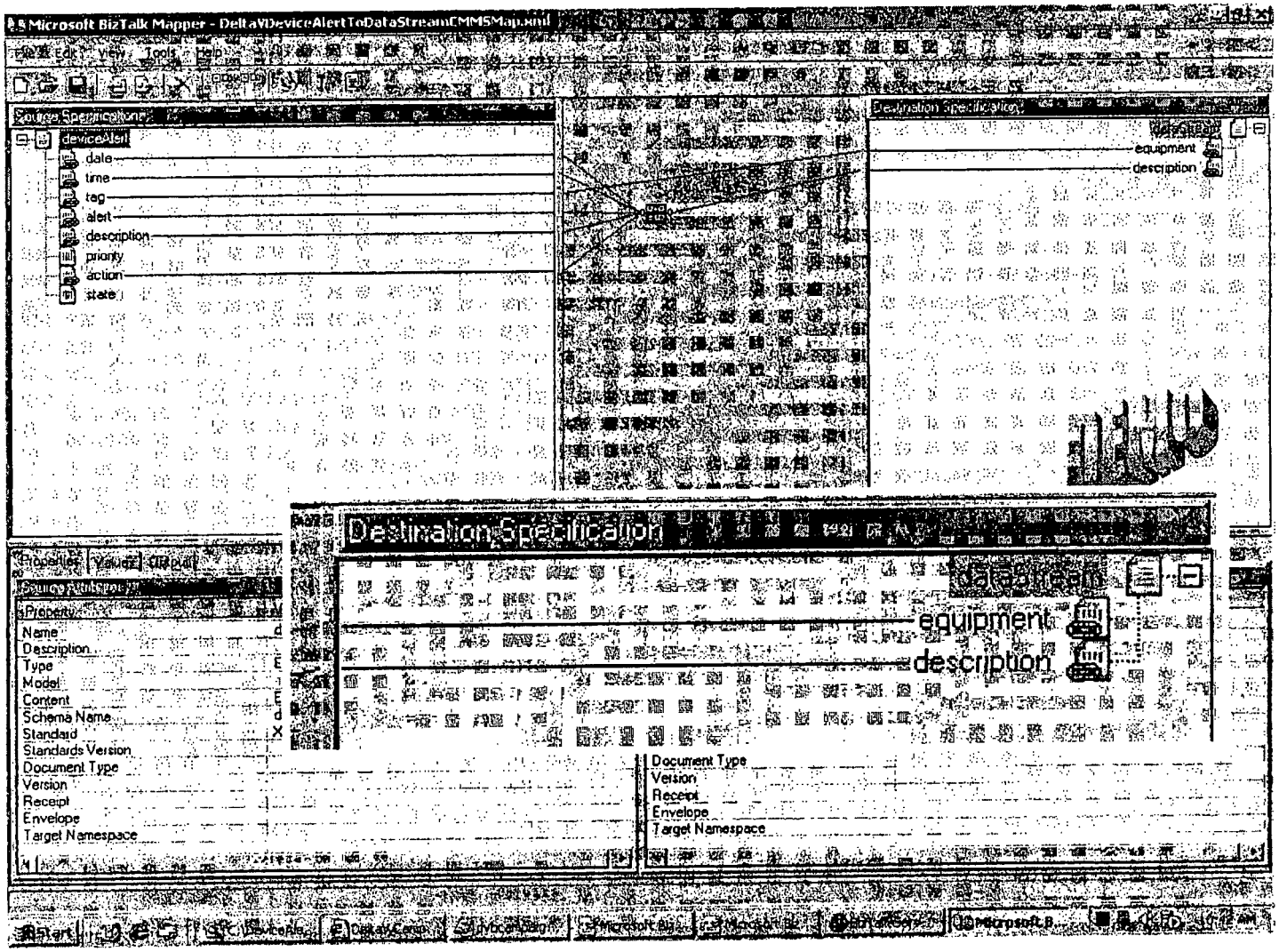
REDACTED

REDACTED

REDACTED


Microsoft BizTalk Mapper - DeltaYDeviceAlertToDataStreamCMMSMap.xml
deviceAlert
date
time
tag
alert
description
priority
action
state
equipment
description
Properties
Values
Output
Name
deviceAlert
Description
Type
Element
Model
Content
Empty
Schema Name
deviceAlert
Standard
XML
Standards Version
Document Type
Version
Receipt
Envelope
Target Namespace
dataStream
Start


Microsoft BizTalk Mapper - DeltaVDeviceAlertToDataStreamCMMSMap.xml
Source Specification
deviceAlert
date
time
tag
alert
description
priority
action
state
Destination Specification
equipment
description
Source Specification
deviceAlert
date
time
tag
alert
description
priority
action
state


Microsoft BizTalk Mapper - DeltaVDeviceAlertToDataStreamCMMSMap.btm
deviceAlert
date
time
tag
alert
description
priority
action
state
Destination Specification
dataStream
equipment
description
Destination Specification
dataStream
equipment
description
Property
Name
Description
Type
Model
Content
Schema Name
Standard
Standards Version
Document Type
Version
Receipt
Envelope
Target Namespace
Document Type
Version
Receipt
Envelope
Target Namespace

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

ENTERPRISE OPTIMIZATION IS HERE AT LAST

– ANSWERING TODAY'S PROBLEMS WITH OPC, XML, BIZTALK, AND EASY-IT REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

There's little wonder why the Internet and Web are so desirable. An enterprise with facilities all over the world, plus its customers and suppliers, can be linked at high speed at miniscule cost. The new kid on the corporate communications block will be knocking EDI for a loop. To accommodate burgeoning Web-based communications, efficient business-to-business (B2B) and business-to-consumer (B2C) e-Commerce methodologies are being developed, and ERP and manufacturing execution systems (MES) suppliers are introducing e-versions of their products. Real and *de facto* standards continue growing and assisting -- Ethernet, TCP/IP, HTTP, HTML, XML, Java, JINI, Active X, OPC, BizTalk, etc.

REDACTED

REDACTED

REDACTED

REDACTED

REDACTED

Spreading the Alert Without Human Intervention

REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED

REDACTED ERP systems crave event-driven transactional data. Typical might be the scenario shown in Figure 6. An intelligent field device, such as a Foundation™ fieldbus flow transmitter, detects that it has a problem, such as blocked impulse lines. The transmitter sends a message -- known as a Device Alert -- to the process automation system controller to indicate that a maintenance condition has arisen. (The message may also contain recommended actions as well as the transmitter's license plate, maintenance history, etc).

REDACTED

Ideally, the above alert would be forwarded as well to a computerized maintenance management system (CMMS) to automatically generate a work order. If urgent, the alert might also be transmitted to a maintenance technician's pager or mobile phone for immediate attention.

Should the transmitter need to be replaced based on the type of alert reported (i.e., Failed), the message could be sent over the Internet to the ERP system at the head office and interfaced with

the ERP's Inventory Control module to check availability of replacement parts. If necessary, the field device could then interface with the ERP's Order Processing module to place an order for new parts. The ERP's accounting system would issue a purchase order, again via the Internet, to a supplier for replacement parts.

Now that's enterprise optimization! The transmitter realizes it has a problem and sets in motion a chain of actions that reports the problem to the appropriate plant staff, generates a work order to correct the problem, notifies the maintenance staff if immediate action is required, checks inventory to see if spare parts exist, and places an order for new parts if necessary. The chain can even add the new device to the maintenance system database, auto-commission the replacement device and auto-record the installation in the maintenance log. All completed without human intervention.

REDACTED

The Internet Language XML Speaks Up

So how will the intelligent field device in the example above send its device alert out over the Internet? Routing a message over the Internet requires either the familiar Hyper Text Markup language (HTML) or the Extensible Markup language (XML) touched on earlier. HTML contains standard pre-defined ISO markup tags that describe only the presentation of each block of type or art on a web page -- type face, type size, type width, bold, italic, color, positioning, etc. HTML presents data designed for human -- not computer -- reading. Typical of HTML messages are most of today's web pages.

Although HTML may be satisfactory for sending process control warnings and instructional copy to a web browser, it's minimally useful for the more intelligent communications required by ERP systems, such as the foregoing inventory verification and order processing example. HTML cannot describe the complex data required by the electronic consumers of plant information. Data presentation is one thing, integration is another.

This is where XML fits in. Released in 1998, platform-independent XML provides the needed self-describing structure. This open and vendor-neutral standard language permits user-defined custom markup tags to be developed for each element (field) of variable information, such as date, name, title, company, address, city, zip, and phone. XML also permits a nested hierarchy of elements and their data types. XML-based plant event data can be integrated with the company ERP system and searched, read, displayed, and manipulated as desired. Also, because XML is related to HTML, XML can be inserted within the boilerplate of an HTML document for viewing in a web browser.

Wrapped in a Schema

Figure 7 illustrates the device alert shown in Figure 6, but configured as an XML message. Each element is identified by kind. The overall wrapper for an XML message is called a schema. Each message type -- such as the device alert -- has its own schema. The schema provides the required structure for the XML data. The schema also defines the contents of the XML data, including the name and type of each element or attribute. A large number of standard schemas are being developed by various standards bodies, trade organizations, and manufacturers to facilitate XML based process control-to-Internet transactional communications.

The schema and the XML document are actually separate files, with the schema referenced in the XML document. An XML parser contained within XML processing software reads the XML document, finds the reference to the schema, and verifies if the document is valid before the XML processing software makes use of it. The schema validates the document content and determines whether a document is a valid instance of the format defined by that schema. The

schema is also useful in that it describes the data for use by others or other computing applications.

As an indication that XML is here to stay, Microsoft had included an XML parser in its Internet Explorer Version 5.0 browser. This allows the viewing of XML files on any PC running IE 5.0 simply by launching the XML file. In addition, since XML files are text based, they can be read using a simple text editor such as Notepad packaged with the Windows operating system.

REDACTED

Are We There Yet?

XML and schemas get us only half the way toward exchanging transactional data between enterprises -- or even within an enterprise -- via the Internet. One reason is that most ERP systems operational today employ EDI, delimited, or flat file documents sent over Value Added Networks (VANs). So we're back to the incompatibility question again.

Second, even for business partners initiating pure XML/Internet B2B document exchanges straight through from back office-to-back office, nothing yet exists in the XML/Internet world like the transaction process standardization developed over the years for EDI. Last, thousands of schemas have been developed since XML was released. If, for example, the schemas for your company's purchasing related documents differ from those of your vendors, can these schemas be transformed to transparently mesh with one another?

REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED Well, bridges of a different type – call them XML transaction servers -- are coming to the rescue for Internet communications to facilitate the transfer of transactional data from one system to another. These servers will bridge older ERP systems to the Internet and ease the development of pure XML/Internet B2B systems.

Figure 8 shows how an XML transaction server might enhance document exchanges between various servers within and without an enterprise. REDACTED REDACTED REDACTED REDACTED REDACTED REDACTED

REDACTED

REDACTED

In addition to orchestrating the business process, the BizTalk Server provides tools for the easy creation of XML document schemas and the transformation of an inbound schema to an outbound schema. For the later, the BizTalk Mapper application generates Extensible Stylesheet Language Transformation (XSLT) files for transforming documents.

Figures 11 and 12 illustrate how inbound and outbound XML schemas might be mapped for the device alert example. The inbound schema defines the information transmitted by the intelligent field device to the process automation system. The outbound schema defines the information required by the CMMS.

Any time a device alert transaction is sent from the process automation system to theBizTalk Server, the data contained in the device alert is validated against the device alert schema shown in Figure 11. The BizTalk Server can work with nearly any kind of input data as mentioned before; it's simply told what data to expect by the inbound schema definition. If the inbound information is valid, it is mapped into an outbound schema tailored specifically to the CMMS package (Figure 12). The required device alert information is then transported to the CMMS application using whatever transport mechanism is required, for example DCOM or HTTP.

Almost There

To move XML transactional data in and out of a process automation system, a final software application running in a process automation system workstation would be required to translate native process automation system data to XML, and vice-versa. Transactions occurring at the plant level could use this perhaps graphically-configured translation application to send XML based information to the enterprise. Transactions occurring at the enterprise level could also use the application to send XML based information to the plant.

The process automation system must decide what transactional information it wants to make available through this translation application, and also what transactional information it wants to receive, by defining the appropriate schema. The following are typical transactions that a process automation system may want to make with the enterprise.

- Import or export historical data
- Import or export process design configuration data
- Import production schedule data
- Export production data

A process control vendor could package the various technologies and techniques needed to bring an IT/Internet network into the process automation system. This would assure that the unique requirements and characteristics of process control are taken into account in building the XML based transactional data exchanges with the enterprise.

The package might include the bi-directional process-to-XML application mentioned above, XML schema definitions of typical control system transactions, and custom software components to help implement business rules appropriate to process control or even specific to a particular vertical industry within process control (i.e., pharmaceutical, oil and gas, pulp and paper, etc.). It would also make sense to provide the development tools required for an end-user to create his own XML based transactions and schemas to suit his company requirements.

Going for a Test Ride

REDACTED

Here's how the aforementioned flow transmitter device alert might simultaneously notify the operator at his workstation, page a maintenance technician, and generate a CMMS work order.

REDACTED

Figure 15 details the device alert as it might be presented in a Datastream MP2 CMMS work order log. In traveling to the CMMS, the alert is translated from the fieldbus format REDACTED REDACTED to an XML schema, forwarded to the transaction server as an input for validation, mapped to an output schema tailored specifically to CMMS, and relayed to the CMMS package. Figures 16 and 17 show the use of a string concatenation to map five input schema elements into one output element. The output schema thereby conforms with this CMMS maker's specifications, which call for only two database records -- equipment and description.

Figure 18 illustrates the prospective device alert's business process-flow chart and implementation logic created in the transaction server. Note the Microsoft Message Queue (MSMQ) for retrieving the input XML schema plus the use of script to check the priority of the device alert. In this example specially prepared COM objects were applied to define the actions required under this Maintenance Alert. Other objects could have been automatically selected if an Advisory Alert or Fail Alert had been selected manually (as shown in Figure 14).

REDACTED